REMARKS

Reconsideration of the pending application is respectfully requested on the basis of the following particulars:

Objection to the drawings

The Examiner has objected to the drawings stating that Fig. 2 does not show "B-accu" which is discussed in the specification. However, in the last paragraph on page 2 of the specification, several registers and areas of a CPU are listed along with "B-accu BAC", wherein "B-accu" shown to be abbreviated as "BAC", which is shown on Fig. 2 in the third box from the left at the top of the figure. Accordingly, withdrawal of the objection is respectfully requested.

Objections to the abstract

The abstract of the disclosure has been amended eliminate improper legal phraseology, thereby obviating the Examiner's objection. Withdrawal of the objection to the abstract is respectfully requested.

Claim objections

Claim 10 has been amended to include the missing colon after "comprising". The document filepath reference ("S:\producer\...\appendix of claims wpd") has been removed from the list of current claims appearing herein. Accordingly, withdrawal of the claim objections is respectfully requested.

Rejection of claims 1-14 under 35 U.S.C. § 112, first paragraph

Claims 1-14 currently stand rejected as failing to comply with the enablement requirement. This rejection is respectfully traversed for the following reasons.

The Examiner alleges that the specification does not teach how the final check sum, determined at the completion of execution of an instruction, is compared with an initial check sum, determined before the execution of a next instruction is begun. However, referring to the specification at lines 6-9 of the last paragraph of page 3, it is clearly stated that "the initial check sum is compared by a comparer with the final check sum stored in the memory from the previously executed first instruction", and that "[i]n case no manipulation was performed on the CPU, the initial and final check sums match and the value of the result of the comparison is zero." It is respectfully submitted that a person skilled in the art would clearly recognize that a comparator, a well known component of digital logic circuits, performs exactly this function. In its simplest form, a comparator (or comparer) compares two digital values, such as by a bitwise exclusive-or (XOR) function to produce an output signal that indicates whether or not the values match.

The Examiner argues that it is not understood how the CPU loads the initial check sum formed in parallel to loading of a second instruction. It is respectfully submitted that this reveals a misunderstanding by the Examiner of the function of the present invention. The CPU does not load the initial check sum. Rather, "the initial check sum is formed [...] parallel to the loading of the second instruction." Loading of an instruction is a fundamental process of instruction execution by a processor, a concept known generally as "von Neumann architecture" since 1946 and used in virtually all present computer processors including the Intel 8051 processor which exemplifies a typical computer processor in Fig. 1 of the present application.

Although not depicted in Fig. 1, a person skilled in the computing arts would know that a typical computer processor includes a program counter and an instruction register. The instruction register holds the instruction currently under execution, and the program counter holds the address in a program memory of the next instruction to be executed. After completion of execution of a first instruction, a next instruction referenced by the program counter is loaded from the program memory into the instruction register to be executed next.

Along with this understanding of the basic process, in the processor, of first executing and instruction and then loading a next instruction, a person of skill in the

computing arts would understand that the phrase "formed [...] parallel to the loading [...]" means that the check sum of the registers of interest is performed while the processor is loading the instruction register in preparation for execution of the next instruction. A hardware solution for determining a check sum value based on contents of the processors registers is suggested in the specification, from the last paragraph of page 2 through the first full paragraph of page 3, wherein hardware logic elements in communication with the processor registers of interest produce a check sum independently from the processor's work. While the Examiner states that "it is unclear where the register content [...] resides", it is submitted that it is eminently clear that the register content resides within the registers.

The Examiner continues, stating that "the only way that is apparent for invention enablement is simply calculating the check sum twice in a row. However, the invention clearly states the purpose is to prevent manipulation [...]." The implication of the Examiner's argument, that the function of preventing manipulation is not achieved by "simply calculating the check sum twice in a row", rests on an assumption that manipulation of the processor registers is not possible between the end of execution of a first instruction and the start of execution of a next instruction. This assumption, however, is flawed. Referring to the last paragraph of page 4 of the specification, it is made clear that in at least one embodiment a relatively long time period may elapse between instructions. Especially in the case of a smart card, as well as other settings, continued operation of the processor is not always expected. Thus, manipulation of the CPU may occur when the processor is not running. In such a case, the manipulation is easily detected by comparison of the successive check sum values once operation of the CPU recommences at the "next" instruction.

Regarding the Examiners argument that the "term instruction introduces additional level of ambiguity since instruction to the processor may be necessary to calculate the checksum", it is respectfully submitted that, while the specification does make reference to a possible software implementation of the check sum calculation, the term "instruction" is used consistently and unambiguously throughout the specification in the context of the individual, CPU-level instructions, or opcodes, executed sequentially within the CPU

during the ordinary course of operation of the CPU, and not to other computer programming statements or the like that might be employed or executed during the course of a software algorithm directed to a check sum calculation.

It is respectfully submitted that, for at least these reasons, the claims of the present application are fully enabled. Accordingly, withdrawal of the rejection is requested.

Rejection of claims 1-14 under 35 U.S.C. § 112, second paragraph

Claims 1-14 currently stand rejected as being indefinite for failing to particularly point out and distinctly claim the subject matter regarded as the invention. This rejection is respectfully traversed for the following reasons.

The Examiner finds the terms "the clock cycles" in claim 2, and "the clock signal supply" in claim 3, to be lacking antecedent basis. Amendments to claims 1 and 2 serve to obviate this finding.

In claim 1, the Examiner finds the term "and stored" to be unclear regarding what is stored. Claim 1 has been amended to clarify that it is the final checksum that is stored.

The Examiner states that the term "instruction" in claim 1 is not understood because it is unclear whether the term refers to "primitive processor's instructions, assemble language instructions, some high-level language instructions or something else." This question is regarded as irrelevant to the inventive principle since within the CPU each "instruction" is seen as a single, or a series of, atomic machine instructions regardless of whether the term "instruction" is taken to be an atomic machine instruction (or primitive processor instruction), or an instruction or programming statement according to a higher-level language. In a higher-level language, an instruction or programming statement often becomes simply a longer sequence of atomic machine instructions. Thus, it is possible that the final and initial check sums are compared between two atomic machine instructions or between two sequences of atomic machine instructions. Nonetheless, it is respectfully submitted that the term "instruction" is sufficiently clarified in the 2nd paragraph of page 2 as comprising an opcode, and as such would be construed by a person skilled in the art as an atomic machine instruction.

Regarding the terms "forming" (in claim 1) and "formed" (in claim 10), claims 1 and 10 have been amended by substituting "determining/determined" for "forming/formed." It is respectfully submitted that this expression is clearly understood by a person skilled in the art as referring to producing a check sum from the contents of the registers through calculational, combinatorial, relational, or other operations.

The Examiner states that the language of Claim 3 is unclear as to whether the claim indicates that the computer is powered off or simply that the instruction is being stopped. Referring to the specification at line 8 of the 2nd paragraph on page 2, "the clock [signal] supply is discontinued, for example, so that no further execution of instructions is possible." Thus, the result is that no further execution of instructions is possible. Discontinuing the clock signal is one method to achieve no further execution of instructions. Other methods include causing an interrupt, causing the CPU to reset (see the specification at page 2, lines 9-10 of the second paragraph). Moreover, referring to the specification at page 4, first paragraph, it is further taught that an error "causes an abortion of instruction processing." It is taught that to abort processing of instructions, "the processor can be stopped, a security sensor activated or, in the case of a smart card, the smart card withheld by the terminal." Thus, it is submitted that the language of claim 3 is clear in that both an interrupt and the discontinuance of the clock signal supply cause the desired action taught in the specification.

Regarding the Examiner's rejection of claim 4, claim 4 has been amended to clarify the language regarding deriving the necessary number of clock cycles for an opcode from a logic circuit. It is submitted that it is well known to persons skilled in the computing arts that, in a CPU such as described in the instant application, a system clock synchronizes the execution of instructions. It is well known that individual instructions require a known and fixed (or determinable) number of clock cycles for complete execution. It is also known that different instructions (or opcodes) often require a different number of clock cycles for execution. It is respectfully submitted that the amended language of claim 4 makes clear that a logic circuit is used to derive, for a given opcode, the number of clock cycles required for execution.

Regarding the Examiner's rejection of claim 6, it is respectfully submitted that the specification clearly teaches a distinction between random and defined events. Reference in the specification to a "random event" would be understood to refer to an event that occur randomly in time with no association with other specific event or triggering cause. While a "random event" may indeed be "scheduled" in time according to, for example, a random number generator, it is nonetheless disassociated from any other specific event or triggering cause. In contrast, the specification clearly identifies exemplary "defined events" in association with a specific event or triggering cause. These include a finding of a specific value or content within a given register, the execution of a predetermined number of instructions, and exceeding a defined time threshold between execution of a first instruction and a next instruction. Thus, it is respectfully submitted that a distinction between random and defined events is clearly stated within the specification.

Regarding the Examiner's rejection of claim 7, it is respectfully submitted that the phrase "in time-dependent fashion" would be clearly understood by a person skilled in the art as referring to an event scheduled in time such as an event that occurs at a temporally defined interval (for example, every minute, every 10 microseconds, once daily), at a fixed time-point (1 millisecond after processor execution begins, 10:00 GMT daily), and the like.

Claim 9 has been amended to eliminate the phrase "in each case", thereby obviating the Examiner's rejection.

Conclusion

In view of the amendments to the claims, and in further view of the foregoing remarks, it is respectfully submitted that the application is in condition for allowance. Accordingly, it is requested that claims 1-14 be allowed and the application be passed to issue.

If any issues remain that may be resolved by a telephone or facsimile communication with the Applicant's attorney, the Examiner is invited to contact the undersigned at the numbers shown.

Respectfully submitted,

BACON & THOMAS, PLLC
625 Slaters Lane, Fourth Floor
Alexandria, Virginia 22314-1176
Phone: (703) 683-0500

Date: April 26, 2005

JUSTIN J. CASSELL
Attorney for Applicant
Registration No. 46,205